CS4677 Computer Forensics Live Data Collection

Chris Eagle

Spring '06

Reading

- Text
  - Familiarize yourself with “Case Studies” pages xxv-xxix
  - Chapters 1 & 2

Reading

- Blackhat Presentation
  - http://www.blackhat.com/presentations/bh-usa-03/bh-us-03-willis-c/bh-us-03-willis.pdf
- Secret Service Guidelines
  - http://www.treas.gov/usss/electronic_evidence.shtml
- Curtis Rose Whitepaper
  - Windows Live Incident Response
  - http://web.archive.org/web/20040218163741/http://www.sytexif.com/whitepaper.htm

Tools

- dcfl-dd
  - http://dcfldd.sourceforge.net/
- George Garner's tools
  - http://users.erols.com/gmgarner/forensics/
- Bootable CD's
  - F.I.R.E 0.4a iso
    - http://sourceforge.net/projects/biatchux
  - Knoppix-std
    - http://www.knoppix-std.org/

Incident/Crime Scene

- What to collect?
  - Host based evidence
  - Network based evidence
  - Other evidence
    - Local search for passwords written on post-its etc…
    - Media in file cabinets
    - Connected devices
      - Digital cameras, USB keys

What to Collect?

- If you encounter a live system you may want to gather some evidence before shutting it down
- Every action you take alters the system
  - Minimize the number of steps
  - Collect only what is necessary
    - Volatile information

Volatile Information

- That information which will be lost when a system is powered down
- Volatile storage
- Volatile system information

Volatile Storage

- CPU register values
  - Too volatile, can’t be captured accurately
- Cache memory contents
  - Beyond our reach, but depending on the cache design, may contain information that is not present in RAM
- RAM contents
  - Potential gold mine
- Virtual memory page file contents

Volatile System Information

- Running Processes
- Active network connections
- Open files and associated processes
- Logged in users
- Current system status information
- Much more

Where To Store Volatile Info?

- Options
  - Local hard drive
    - BAD! Modifies subject system!
  - Record by hand
    - BAD! Tedious, error prone, too much info
  - Save to removable media
    - May need high capacity
    - USB helps here
  - Transfer across network

Evidence Storage

- Removable media or network transfer are best options
  - High capacity USB drives are excellent
    - Make sure they are formatted already
    - May change state of system when first connected
  - Network transfer via NFS/Samba shares or use of netcat
    - Must have proper configuration to do NFS/Samba
    - Can send virtually anywhere with netcat

Evidence Protection

- Set the immutable bit on each of your evidence files to prevent changes

```
chmod ugo-w <file>
```

  - Is not sufficient!
- Use the chattr command to make a file immutable

```
chattr +i <file to protect>
```

Evidence Integrity

- Each tool you run to collect evidence will generate output
- This output needs to be captured
- A cryptographic hash needs to be computed on the saved output
  - ~~md5sum~~
  - ~~sha1sum~~
  - Perform both or use a stronger hash such as SHA-256

Evidence Integrity

- Some tools tailored specifically for forensics will report the md5 hash value for output they have generated
  - You confirm by hashing the saved output file
- MD5 and SHA1 may be broken! Use both or a stronger hash such as SHA-256

Time Stamping

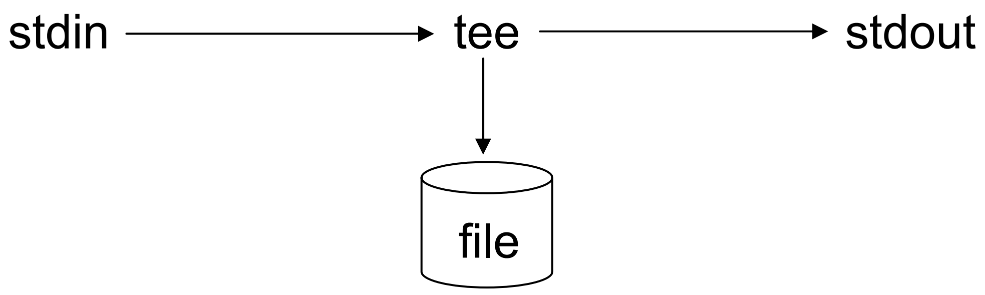
- Consider obtaining cryptographically signed time stamps of your hash values
- Proves that the hash values were obtained no later than a specific time
- One such service is offered at
  - http://www.itconsult.co.uk/stamper/stampinf.htm

Command Plumbing

- A solid understanding of piping and redirection is helpful
  - Redirection
    - > - send standard output of a program to a file (replaces console)
    - < - take standard input for a program from a file (replaces keyboard)
  - Piping
    - | - connect standard output of one program to standard input of another program

tee

- It is a good goal to attempt to generate hash values at the same time you collect a piece of evidence
  - Though not always possible
- The tee command splits its standard input to a file and standard out



tee Examples

netstat –an
- stdout only

netstat –an > netstat.txt
- File only

netstat –an | tee netstat.txt
- Both stdout and file

netstat –an | tee netstat.txt | sha1sum
- File and a sha1sum to stdout

netstat –an | tee netstat.txt | sha1sum > netstat.sha1
- File and a sha1sum to file (nothing to stdout)

Netcat Basics

- Two modes
  - Client mode
    - Copies stdin to a remote “server”
    - Reads from server to stdout
    - Specify IP and port to connect to
  - Server mode
    - Awaits an incoming connection and copies incoming client data to stdout
    - Copies stdin to client computer
    - Specify listen mode and port to listen on

Netcat Example

- Client side – victim machine
  - Pipe output of evidence gathering command into netcat in client mode

    ```
    netstat –anp --inet | nc data.forensics.org 1234
    ```

- Server side – evidence collection machine
  - Receive remote data and save to evidence file

    ```
    nc –l –p 1234 > netstat.txt
    ```

  - Caution switches for nc vary by OS
- Forensics versions of netcat can compute md5 sums as they transfer data

Netstat Example

- Using tee it is possible to compute hash values on receiving side

```
nc -l -p 1234 | tee netstat.txt | sha1sum > netstat.sha1
```

  – Received data is saved to an evidence file and integrity value of that file is computed and saved to a separate file

Cryptcat

- Cryptographically enhanced version of netcat
  - http://sourceforge.net/projects/cryptcat
  - Supply password on the command line to encrypt all traffic using twofish
  - Useful if sending evidence across internet for example

Tools Collection

- Have a collection of tools handy for collecting volatile evidence
  - F.I.R.E is not a bad start
- DO NOT rely on any software on the subject system
  - Bring your own command shells
  - Use statically linked binaries
    - Don’t trust dynamic libraries on subject system
    - F.I.R.E provides this

Getting Started

- Open a clean command shell
  - One you brought with you
  - Statically linked
  - Issue all commands from within this shell

Imaging RAM (text pg. 26)

- Should probably perform early before you make too many changes
- Must have root/Administrator access
- Windows
  - Use George Garner's version of `dd`
- Unix
  - Use standard `dd` or `dcfldd` command

dd Basics

- dd is a standard Unix tool for copying blocks of data
  - Can perform some data conversions
    - But we won’t require these
  - By default copies stdin to stdout
  - Input/output can alternately be a file or device
    - Device can be a single partition or an entire hard drive
  - Can take any portion from any offset of the input
    - Use skip, count, bs options

Imaging RAM w/ dd

- Windows
  - Assuming g is our own usb drive on the victim system

```
dd if=\\.\PhysicalMemory
 conv=noerror,sync bs=4096
 of=g:\ram.img --md5sum
```

  - Imaging across the network

```
dd if=\\.\PhysicalMemory
 conv=noerror,sync bs=4096 --md5sum |
 nc data.forensics.org 1234
```

Imaging RAM w/ dd (II)

- Unix
  - Useful devices
    - /dev/mem – physical memory
    - /dev/kmem – kernel virtual memory
  - Assuming /mnt/usb is our own usb drive on the victim system

```
dd if=/dev/mem conv=noerror,sync bs=4096
  of=/mnt/usb/ram.img
```

```
dcfldd if=/dev/mem conv=noerror,sync bs=4096
  of=/mnt/usb/ram.img hashwindow=0
```

  - Imaging across the network

```
dd if=/dev/kmem conv=noerror,sync bs=4096 | nc
  data.forensics.org 1234
```

```
dcfldd if=/dev/mem conv=noerror,sync bs=4096
  hashwindow=0 | nc data.forensics.org 1234
```

MAC Times

- File MAC times
  - On Unix and NTFS file systems, all files have not one but three time stamps associated with them
    - (M)odification - mtime
    - (A)ccess - atime
    - (C)hange - ctime
  - The dir and ls commands only show mtime by default

System Time

- Current system date and time
  - Windows
    - `date /t > date.txt`
    - `time /t > time.txt`
  - Unix
    - `date > date.txt`

Network Configuration

- Network Interface Configuration
  - Windows
    - `ipconfig /all`
    - IP/MAC address among others
  - Unix
    - `ifconfig -a`
    - `cat /etc/hosts`
    - `cat /etc/resolv.conf`
    - `route`
    - Can indicate if an interface is in promiscuous mode

Active Network Connections

- Current Network Connections
  - Windows
    - `netstat -anobv`
      - a: all listening ports
      - n: numeric display of ports/ips
      - o: owning process id (XP only)
      - b: display executable involved in creating socket
    - Fport - http://www.foundstone.com/
      - Select "Resources" then "Free Tools"
    - TcpView – http://www.sysinternals.com/Utilities/TcpView.html
      - Tcpvcon – console version
        - » tcpvcon -an

Active Network Connections

- Current Network Connections
  - Unix
    - `lsof`
      - List of open files, some of which are network sockets
    - `netstat –anp --inet`

      -a : show all sockets, not just established

      -n : don’t resolve names

      -p : process id/name

      --inet : IP sockets only (no unix sockets)

      CAUTION options vary by OS, those above work with Linux

Recent Network Connections

- Book recommends
  - Windows
    - NetBios table stats

      `nbtstat -c`
    - Current arp table

      `arp -a`
  - Unix
    - `arp`

Users

- Current System User's
  - Windows
    - `psloggedon`
      - http://www.sysinternals.com/ntw2k/freeware/pstools.shtml
  - Unix
    - `who`
    - `w`

Routing Tables

- Windows
  - route print
  - netstat -r
- Unix
  - route -n
  - netstat -rn

Active Processes

- Currently Running Processes
  - Windows
    - `tasklist`
    - `pslist`
    - `psservice`
      - http://www.sysinternals.com/ntw2k/freeware/pstools.shtml
    - ProcessExplorer
      - http://www.sysinternals.com/Utilities/ProcessExplorer.html
  - Unix
    - `ps -aux`
    - Displays any running “services”

Services

- A service is a process
  - Started automatically at system startup
  - Runs in the background to provide various services
- Windows
  - Services must be installed and registered as services
  - Do not necessarily appear under task manager
- Unix
  - Similar idea, but no standard startup across Unix versions
  - RedHat: chkconfig
  - Others: `/etc/init.d`, `/etc/rc.sysinit`, `/etc/rc.local`

Currently Open Files

- Open files
  - Applications can have many files open at any given time
  - Unix extends the notion of a “file” to many different things including network sockets
  - May be the only way to see some “deleted” files
    - A running program can “unlink” a file, effectively deleting it in the eyes of the file system
    - Not officially deleted until the program exits

Open Files

- Listing Open Files
  - Windows
    - `openfiles /Query`
      - `openfiles /Local ON`
        - » enables listing of local open files AFTER REBOOT
        - » Adds performance overhead
    - `psfile`
      - Lists files opened remotely
    - `listdlls`
      - List dlls in use by each executing process
      - http://www.sysinternals.com/ntw2k/freeware/listdlls.shtml
  - Unix
    - `lsof`
    - List open files
    - Lists all files opened by a process including data files and sockets

Drive/Partition Information

- Mounted partitions
  - Windows
    - `volume_dump`
      - George Garner
    - Listed as part of `psinfo`
  - Unix
    - `mount`
      - Lists partition name, mount point, and file system type
    - `fdisk -lu <device>`